Rayleigh quotient methods for estimating common roots of noisy univariate polynomials

Alwin Stegeman and Lieven De Lathauwer

Group Science, Engineering and Technology, KU Leuven – Kulak,
E. Sabbelaan 53, B-8500 Kortrijk, Belgium,
and
Department of Electrical Engineering (ESAT), KU Leuven,
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium,
alwin.stegeman@kuleuven.be, lieven.delathauwer@kuleuven.be

Abstract

The problem is considered of approximately solving a system of univariate polynomials with one or more common roots and its coefficients corrupted by noise. The goal is to estimate the underlying common roots from the noisy system. Symbolic algebra methods are not suitable for this. New Rayleigh quotient methods are proposed and evaluated for estimating the common roots. Using tensor algebra, reasonable starting values for the Rayleigh quotient methods can be computed. The new methods are compared to Newton-Raphson, solving an eigenvalue problem obtained from the generalized Sylvester matrix, and finding a cluster among the roots of all polynomials. In a simulation study it is shown that Newton-Raphson and a new Rayleigh quotient method perform best, where the latter is more accurate when other roots than the true common roots are close together.

Keywords: Rayleigh quotient iteration, univariate polynomials, common roots, noisy polynomials, numerical polynomial algebra.

AMS subject classifications: 13P15, 15A18, 65F15.

Abbreviated title: Rayleigh quotient methods for common roots of polynomials

Research supported by: (1) Research Council KU Leuven: C1 project c16/15/059-nD; (2) the Belgian Federal Science Policy Office: IUAP P7 (DYSCO II, Dynamical systems, control and optimization, 2012-2017)

1 Introduction

Polynomial systems occur in many applications and research fields, most notably in applied mathematics and engineering, e.g. [1, 3, 5]. Often the coefficients are obtained from data and, hence, are not known exactly. Applying symbolic algebra methods is inappropriate and may yield misleading results. Hence, numerical methods to solve noisy polynomial systems are needed. Hybrid methods were introduced by Lazard [12, 13] and Stetter [18, 19], which need a Gröbner basis as input. Recently, the work of Sylvester [20] and Macaulay [14, 15] was combined with numerical linear algebra tools, resulting in purely numerical methods for solving polynomial systems [2, 6].

In this paper we consider a system of univariate polynomials having one or several common roots. The goal is to estimate the common roots when observing a noisy version of the system. Let

$$p_i(x) = c_{d,i}\, x^d + c_{d-1,i}\, x^{d-1} + \cdots + c_{1,i}\, x + c_{0,i}\,, \qquad i = 1, \ldots, n\,, \tag{1.1}$$

with real or complex coefficients $c_{j,i}$. The degree of the polynomials may differ (i.e., we allow for $c_{d,i} = 0$), but the maximal degree equals d (i.e., $c_{d,i} \neq 0$ for some i). Let the degree of $p_i(x)$ be d_i. We assume there exist m simple common roots, i.e., $p_i(r_h) = 0$ for $i = 1, \ldots, n$, $h = 1, \ldots, m$. We focus on numerical methods for estimating the common roots r_h when (1.1) is observed with coefficients corrupted by noise.

We propose a new numerical methods for estimating the common roots of (1.1) and compare it in a simulation study to existing methods. The new methods are generalizations of Rayleigh quotient iteration for finding eigenpairs of a square matrix. The roots of each polynomial $p_i(x)$ are eigenvalues of its companion matrix $\mathbf{C}_i$, with associated eigenvectors of Vandermonde form with as generators the roots. Our new methods find common eigenpairs of $\mathbf{C}_1, \ldots, \mathbf{C}_n$ using the least squares framework of Rayleigh quotient iteration. For matrices ($n = 1$) it is known that Rayleigh quotient iteration may produce false results when the starting value is taken random. However, for $d_i = d$ the n companion matrices can be arranged in a $d \times d \times n$ tensor and an orthogonal transformation of this tensor, which implies an orthogonal transformation of the least squares problem, allows us to rationally choose a starting value for the common eigenvector, which dramatically improves estimation accuracy. This approach does not work in the case of $n = 1$ polynomial.

As existing methods we consider Newton-Raphson, solving an eigenvalue problem obtained from the generalized Sylvester matrix [6], and estimating the common roots by finding a cluster among the roots of all $p_i(x)$. The latter method yields an estimate of a common root as the mean of the obtained cluster. We use their estimates as starting values for Newton-Raphson. In an extensive simulation study various combinations of maximal degree d, individual degrees d_i,

number of polynomials n, complex and/or real roots, proximity of roots other than the common roots, and sizes of the derivatives at the common roots are considered, and the performance of the methods is compared.

The considered rootfinding methods, including ordinary Rayleigh quotient iteration, are well-known methods for computing the roots of one noisefree univariate polynomial, see e.g. [16]. However, to the authors' knowledge the problem of computing common roots of noisy univariate polynomials has not been studied before.

This paper is organized as follows. In section 2 we described the exisiting numerical methods we consider and introduce three variants of our Rayleigh quotient approach. In section 3 we derive properties of local and global minima in our least squares Rayleigh quotient framework, and provide an illustrative example with a local minimum. Section 4 contains the results of our simulation study. Finally, section 5 summarizes our findings and provides additional discussion.

As notation we use the following. We denote scalars, column vectors, matrices, and higher-order tensors as x, $\mathbf{x}$, $\mathbf{X}$, and $\mathcal{X}$, respectively. The transpose of $\mathbf{X}$ is denoted as $\mathbf{X}^T$, and the complex conjugate transpose (or Hermitian transpose) as $\mathbf{X}^H$. The inverse of a square matrix is denoted as $\mathbf{X}^{-1}$. An all-zero vector is denoted as $\mathbf{0}$. We denote the Frobenius norm as $\|\cdot\|$, which is defined as $\|\mathcal{X}\| = (\sum_{i,j,k} |x_{ijk}|^2)^{1/2}$ and analogously for matrices and vectors. For an $I_1 \times I_2 \times I_3$ tensor $\mathcal{X}$ we denote its multilinear multiplication by matrices $\mathbf{S}$ ($J_1 \times I_1$), $\mathbf{T}$ ($J_2 \times I_2$), and $\mathbf{U}$ ($J_3 \times I_3$) as $\mathcal{Y} = (\mathbf{S}, \mathbf{T}, \mathbf{U}) \cdot \mathcal{X}$, where $\mathcal{Y}$ has size $J_1 \times J_2 \times J_3$ and entries $y_{pqr} = \sum_{ijk} s_{pi}\, t_{qj}\, u_{rk}\, x_{ijk}$.

2 Numerical methods

Here we discuss in detail the numerical methods used to estimate the common roots of the noisy polynomial system (1.1). We start with existing methods in sections 2.1 - 2.3, and discuss the new Rayleigh quotient iteration method in section 2.4.

2.1 Newton-Raphson

The classical numerical method for solving (1.1) is Newton-Raphson. Let $F(x) = (p_1(x) \ \ldots \ p_n(x))^T$ and $\nabla_F(x) = (\frac{\partial p_1}{\partial x}(x) \ \ldots \ \frac{\partial p_n}{\partial x}(x))^T$. Newton-Raphson updates are obtained by least squares solving of $\nabla_F(x_k)\,(x_{k+1} - x_k) = -F(x_k)$, which yields

$$x_{k+1} = x_k - (\nabla_F(x_k)^H \nabla_F(x_k))^{-1}\, \nabla_F(x_k)^H F(x_k)\,. \tag{2.1}$$

It is well-known that Newton-Raphson has quadratic convergence when started close enough to a common root, for the system without noise [8]. A method for obtaining starting values is discussed

in section 2.3. Newton-Raphson can be used to estimate one common root only and the chosen starting value will determine to which common root the algorithm converges.

2.2 Eigenproblem from generalized Sylvester matrix

A classical result for finding common roots of two univariate polynomials involves the Sylvester matrix. Let $\mathbf{c}_i^T = (c_{0,i}\ c_{1,i}\ \ldots\ c_{d_i,i})$ for $i = 1, 2$, and let

$$\mathbf{B}(\mathbf{c}_i^T, s) = \begin{bmatrix} \mathbf{c}_i^T & 0 & 0 & \ldots & 0 \\ 0 & \mathbf{c}_i^T & 0 & \ldots & 0 \\ 0 & 0 & \mathbf{c}_i^T & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \ldots & 0 & \mathbf{c}_i^T \end{bmatrix}, \tag{2.2}$$

have size $s \times (d_i + s)$. The $(d_1 + d_2) \times (d_1 + d_2)$ Sylvester matrix equals

$$\mathbf{S} = \begin{bmatrix} \mathbf{B}(\mathbf{c}_1^T, d_2) \\ \mathbf{B}(\mathbf{c}_2^T, d_1) \end{bmatrix}. \tag{2.3}$$

Sylvester [20] proved that $p_1(x)$ and $p_2(x)$ have a common root if and only if $\det(\mathbf{S}) = 0$; see also [3]. Let $\mathbf{y} = \mathrm{Vdm}(r, k)$ denote the Vandermonde vector $\mathbf{y} = (1\ r\ r^2\ \ldots\ r^{k-1})^T$. If r is a common root of $p_1(x)$ and $p_2(x)$, then $\mathbf{S}\,\mathbf{y} = \mathbf{0}$ for $\mathbf{y} = \mathrm{Vdm}(r, d_1 + d_2)$. For m simple common roots the nullity of $\mathbf{S}$ equals m and there are m linearly independent Vandermonde vectors in its null space.

It is shown in [6] that the common roots may be obtained from the null space of $\mathbf{S}$ via an eigendecomposition. Let $\mathbf{N}$ ($(d_1 + d_2) \times m$) be a basis of the null space of $\mathbf{S}$ (i.e., $\mathbf{S}\,\mathbf{N} = \mathbf{O}$). Then $\mathbf{Y} = \mathbf{N}\,\mathbf{T}$ contains the Vandermonde basis of the null space of $\mathbf{S}$, for some nonsingular $\mathbf{T}$. Define the row selection matrices $\mathbf{R}_1 = [\mathbf{I}_{d_1+d_2-1}\ \mathbf{0}]$ and $\mathbf{R}_y = [\mathbf{0}\ \mathbf{I}_{d_1+d_2-1}]$. Using the shift property of the Vandermonde vectors, we obtain $\mathbf{R}_1\,\mathbf{Y}\,\mathbf{D} = \mathbf{R}_y\,\mathbf{Y}$, with $\mathbf{D}$ the diagonal matrix containing the m common roots. This can be rewritten as $\mathbf{T}\,\mathbf{D}\,\mathbf{T}^{-1} = (\mathbf{R}_1\,\mathbf{N})^+\,\mathbf{R}_y\,\mathbf{N}$, where the right-hand side can be computed and the left-hand side is an eigendecomposition. The common roots can thus be obtained as the eigenvalues in $\mathbf{D}$.

For more than two polynomials ($n > 2$) this approach can also be used [6]. For degree d_s and n polynomials the generalized Sylvester matrix of size $\sum_i (d_s - d_i + 1) \times (d_s + 1)$ is defined as

$$\mathbf{S}(d_s) = \begin{bmatrix} \mathbf{B}(\mathbf{c}_1^T, d_s - d_1 + 1) \\ \vdots \\ \mathbf{B}(\mathbf{c}_n^T, d_s - d_n + 1) \end{bmatrix}. \tag{2.4}$$

Analogous to the $n = 2$ case, a common root r yields a vector $\mathbf{y} = \mathrm{Vdm}(r, d_s)$ in the null space of $\mathbf{S}(d_s)$. For d_s large enough the nullity of $\mathbf{S}(d_s)$ equals the number m of common roots and they can be obtained via an eigendecompositon as explained above. Note that we need $\mathbf{S}(d_s)$ to be square or tall for the method to work, which implies that d_s needs to satisfy

$$(n-1)\,(d_s+1) \geq \sum_{i=1}^{n} d_i\,. \tag{2.5}$$

See [2] for the multivariate generalization of the theory, which uses the Macaulay matrix instead [14] [15] and forms an alternative to eigendecomposition methods using a Gröbner basis as input [12] [13] [18] [19].

To apply this method for noisy polynomial systems, we either need to estimate the rank of $\mathbf{S}(d_s)$, or specify the expected number of common roots m in advance. We then use the right-singular vectors corresponding to the m smallest singular values to form a basis $\mathbf{N}$ for its null space.

2.3 Computing roots of single polynomials

Since a common root is a root of each polynomial $p_i(x)$, we may also try to estimate a common root by finding a cluster among the roots of all noisy $p_i(x)$. We use the following approach to obtain this cluster. In total there are $\sum_i d_i$ roots and we can form $\prod_i d_i$ sets of n roots, one of each polynomial $p_i(x)$. Of each such set we compute the standard deviation and we take the mean of the set with the smallest standard deviation as estimate of a common root. Since checking all sets of n roots may take too much time, we use the following heuristic. For each root of $p_1(x)$ we create a set of n roots by including the root of $p_i(x)$ that is closest to the root of $p_1(x)$, for $i = 2, \ldots, n$. We then obtain an estimate of common root as stated above. Note that this method estimates only one common root. The heuristic method is fast to execute and we use it to obtain starting values for the Newton-Raphson algorithm. For large n and mean-zero noise on the coefficients of the polynomials, the estimate of the common root will be very close to the true common root by the law of large numbers.

One may expect that a classical clustering method such as k-means can also be used. However, clustering methods aim to assign all roots to a cluster while we only need to cluster the noisy common roots. Moreover, the restriction of including one root of each polynomial in the cluster is dropped. For many noisefree examples k-means clustering fails to estimate the common root accurately due to other roots close to the common root being included in the same cluster. Also for noisy examples k-means clustering provides less accurate estimates of the common root than the heuristic approach above.

2.4 New Rayleigh quotient methods

For simplicity we first assume that $d_i = d$ for all i. Let $\mathbf{C}_i$ be the $d \times d$ companion matrix associated with polynomial $p_i(x)$, which we define as

$$\mathbf{C}_i = \begin{bmatrix} -\frac{c_{d-1,i}}{c_{d,i}} & -\frac{c_{d-2,i}}{c_{d,i}} & \cdots & -\frac{c_{1,i}}{c_{d,i}} & -\frac{c_{0,i}}{c_{d,i}} \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ & \ddots & \ddots & 0 & 0 \\ & & 0 & 1 & 0 \end{bmatrix} . \tag{2.6}$$

Then a common root r is a common eigenvalue of all $\mathbf{C}_i$ with common associated eigenvector $\mathbf{y} = \mathrm{Vdm}(r^{-1}, d)$. Let

$$\mathbf{M}(x) = \begin{bmatrix} \mathbf{C}_1 \\ \vdots \\ \mathbf{C}_n \end{bmatrix} - x \begin{bmatrix} \mathbf{I}_d \\ \vdots \\ \mathbf{I}_d \end{bmatrix} . \tag{2.7}$$

It holds that $\mathbf{M}(r)\,\mathbf{y} = \mathbf{0}$ for a common root r and associated eigenvector $\mathbf{y} = \mathrm{Vdm}(r^{-1}, d)$. Analogous to Rayleigh quotient iteration (RQI) for one matrix we update x for fixed $\mathbf{y}$ by minimizing $\|\mathbf{M}(x)\,\mathbf{y}\|^2$, which yields the update

$$x = n^{-1}(\mathbf{y}^H\mathbf{y})^{-1}\mathbf{y}^H(\mathbf{C}_1 + \cdots + \mathbf{C}_n)\,\mathbf{y}\,. \tag{2.8}$$

Note that (2.8) is the mean of the Rayleigh quotients of each $\mathbf{C}_i$, and identical to the Rayleigh quotient of the mean companion matrix. For the update of $\mathbf{y}$ for fixed x we consider two options. The first option is to apply RQI using the mean companion matrix $\mathbf{C}_\bullet = n^{-1}(\mathbf{C}_1 + \cdots + \mathbf{C}_n)$. Note that a common eigenvector $\mathbf{y}$ of all companion matrices with common eigenvalue r is also an eigenvector of $\mathbf{C}_\bullet$ associated with eigenvalue r. To obtain an RQI algorithm using update (2.8), we apply one inverse iteration using $(\mathbf{C}_\bullet - x\,\mathbf{I}_d)$ to update $\mathbf{y}$ for fixed x. That is, we solve

$$(\mathbf{C}_\bullet - x\,\mathbf{I}_d)\,\mathbf{y}_{k+1} = \mathbf{y}_k\,, \tag{2.9}$$

with $\mathbf{y}_k$ obtained from the previous iteration, followed by norming $\mathbf{y}_{k+1}$ such that $\mathbf{y}_{k+1}^H\mathbf{y}_{k+1} = 1$. We use the abbreviation RQI-M to refer to the algorithm with updates (2.8) and (2.9), with "M" referring to the Mean companion matrix. As will be shown below, in the noisefree case the common roots can be obtained by computing initial values for $\mathbf{y}$ in an appropriate way. Let $\widetilde{\mathbf{C}}_\bullet$ denote the noisy version of $\mathbf{C}_\bullet$, which has eigenvalue $\tilde{r} \approx r$ and eigenvector $\tilde{\mathbf{y}} \approx \mathrm{Vdm}(r^{-1}, d)$. In that case RQI-M applied to $\widetilde{\mathbf{C}}_\bullet$ exhibits cubic convergence, i.e., $|x_{k+1} - \tilde{r}| = O(|x_k - \tilde{r}|^3)$ and

$\|\mathbf{y}_{k+1} - \tilde{\mathbf{y}}\| = O(\|\mathbf{y}_k - \tilde{\mathbf{y}}\|^3)$ as the iteration number k increases, provided the initial value $\mathbf{y}_0$ is close enough to $\tilde{\mathbf{y}}$; see [21].

The second option to update $\mathbf{y}$ is by minimizing $\|\mathbf{M}(x)\,\mathbf{y}\|^2$ for fixed x. The solution is to take $\mathbf{y}$ as the right-singular vector of $\mathbf{M}(x)$ corresponding to its smallest singular value. Computing this singular vector $\mathbf{y}$ can be done via inverse iteration on $\mathbf{M}(x)^H\mathbf{M}(x)$; see [21]. That is, by solving

$$(\mathbf{M}(x)^H\mathbf{M}(x))\,\mathbf{y}_{k+1} = \mathbf{y}_k\,, \tag{2.10}$$

followed by norming $\mathbf{y}_{k+1}$ such that $\mathbf{y}_{k+1}^H\mathbf{y}_{k+1} = 1$. We refer to the algorithm with update (2.8) and one inverse iteration (2.10) as RQ-LS-I. Let $\widetilde{\mathbf{M}}(x)$ denote the noisy version of $\mathbf{M}(x)$. Since the minimizing pair $(r_{\min}, \mathbf{y}_{\min})$ of $f(x,\mathbf{y}) = \|\widetilde{\mathbf{M}}(x)\,\mathbf{y}\|^2$ does not (necessarily) equal an eigenpair of $\widetilde{\mathbf{C}}_\bullet$, we could not prove the cubic convergence of RQI-M. However, it does hold that for fixed x: $\|\mathbf{y}_k - \mathbf{y}_{\min}\| = O(|\lambda_d/\lambda_{d-1}|^k)$, with λ_d and λ_{d-1} denoting the smallest and second smallest eigenvalues of $\widetilde{\mathbf{M}}(x)^H\widetilde{\mathbf{M}}(x)$, respectively; see [21]. Also, from the least squares updates it follows that $|f(r_{\min},\mathbf{y}) - f(r_{\min},\mathbf{y}_{\min})| = O(\|\mathbf{y} - \mathbf{y}_{\min}\|^2)$ as $\mathbf{y} \to \mathbf{y}_{\min}$, and $|f(x,\mathbf{y}_{\min}) - f(r_{\min},\mathbf{y}_{\min})| = O(|x - r_{\min}|^2)$ as $x \to r_{\min}$.

Since the condition number of $\mathbf{M}(x)^H\mathbf{M}(x)$ is the square of the condition number of $\mathbf{M}(x)$, more stable and accurate results may be obtained when using alternating inverse iteration instead [17, 9]. The right-singular vector $\mathbf{y}$ of $\mathbf{M}(x)$ is also the right-singular vector of (square) $\mathbf{R}$ in the QR-decomposition $\mathbf{M}(x) = \mathbf{Q}\,\mathbf{R}$. We proceed by alternatingly solving

$$\begin{bmatrix} \mathbf{R} \\ \mathbf{y}_k^H \end{bmatrix} \mathbf{y}_{k+1} = \begin{bmatrix} \mathbf{z}_k \\ 1 \end{bmatrix} \quad \text{for } \mathbf{y}_{k+1}\,, \qquad \text{and} \qquad \begin{bmatrix} \mathbf{R}^H \\ \mathbf{z}_k^H \end{bmatrix} \mathbf{z}_{k+1} = \begin{bmatrix} \mathbf{y}_{k+1} \\ 1 \end{bmatrix} \quad \text{for } \mathbf{z}_{k+1}\,. \tag{2.11}$$

After solving the first equation $\mathbf{y}_{k+1}$ is normalized such that $\mathbf{y}_{k+1}^H\mathbf{y}_{k+1} = 1$. After solving the second equation $\mathbf{z}_{k+1}$ is normalized such that $\mathbf{z}_{k+1}^H\mathbf{z}_{k+1} = 1$. Note that (2.11) implies the inverse iteration (2.10), followed by norming $\mathbf{y}_{k+1}$. Linear convergence of $\mathbf{y}_k$ and $\mathbf{z}_k$ to the smallest right- and left-singular vectors of $\mathbf{R}$, respectively, is proven by [17]. We refer to the algorithm with update (2.8) and one alternating inverse iteration (2.11) as RQ-LS-A. In general, minimizing $\|\mathbf{M}(x)\,\mathbf{y}\|^2$ may be more robust than using RQI on the mean companion matrix when the polynomials are affected by noise.

Analogous to Newton-Raphson and Rayleigh quotient iteration for one matrix, the new Rayleigh quotient method estimates one common root only and should be provided with good starting values in order to prevent convergence to a local minimum. Next, we discuss how to obtain a starting value for $\mathbf{y}$ by transforming the least squares problem.

The n companion matrices $\mathbf{C}_i$ can be considered as the frontal slices of a $d \times d \times n$ tensor $\mathcal{C}$. Let the $d \times d \times n$ tensor $\mathcal{D}$ have all frontal slices equal to $\mathbf{I}_d$. Using multilinear matrix multiplication,

the minimization of $\|\mathbf{M}(x)\,\mathbf{y}\|^2$ can be reformulated as the minimization of

$$\|(\mathbf{I}_d, \mathbf{y}^H, \mathbf{I}_n)\cdot\mathcal{C} - x\,(\mathbf{I}_d, \mathbf{y}^H, \mathbf{I}_n)\cdot\mathcal{D}\|^2\,, \tag{2.12}$$

over $(x, \mathbf{y})$ subject to $\mathbf{y}^H\mathbf{y} = 1$. Let $\mathbf{Q}$ be an $n \times n$ orthonormal matrix such that $\mathbf{Q}\,\mathbf{1}_n = \begin{pmatrix} \alpha \\ \mathbf{0} \end{pmatrix}$. For example, matrix $\mathbf{Q}$ can be obtained as the transpose of the right-singular vectors of $\mathbf{1}_n^T$. Tensor $(\mathbf{I}_d, \mathbf{I}_d, \mathbf{Q})\cdot\mathcal{D}$ now has frontal slices $2, \ldots, n$ all-zero and frontal slice 1 equal to $\alpha\,\mathbf{I}_d$. Multiplying by $\mathbf{Q}$ in the third mode does not affect the least squares problem in (2.12). Let $\widetilde{\mathcal{C}} = (\mathbf{I}_d, \mathbf{I}_d, \mathbf{Q})\cdot\mathcal{C}$ have frontal slices $\widetilde{\mathbf{C}}_i$, $i = 1, \ldots, n$, which are linear combinations of the original companion matrices. In matrix unfolding form, the problem can now be written as the minimization of

$$\|(\widetilde{\mathbf{C}}_1 - x\,\alpha\,\mathbf{I}_d)\,\mathbf{y}\|^2 + \left\| \begin{bmatrix} \widetilde{\mathbf{C}}_2 \\ \vdots \\ \widetilde{\mathbf{C}}_n \end{bmatrix} \mathbf{y} \right\|^2\,, \tag{2.13}$$

over $(x, \mathbf{y})$ subject to $\mathbf{y}^H\mathbf{y} = 1$. The second term in (2.13) only depends on $\mathbf{y}$ and involves $n-1$ of the n transformed companion matrices. It can be used to obtain an initial value for $\mathbf{y}$ as follows. When one common root exists the second term can be made zero by picking $\mathbf{y}$ from the one-dimensional null space of of $[\widetilde{\mathbf{C}}_2^T \;\ldots\; \widetilde{\mathbf{C}}_n^T]^T$. When m simple common roots exist the second term can be made zero by picking $\mathbf{y}$ from the m-dimensional null space of $[\widetilde{\mathbf{C}}_2^T \;\ldots\; \widetilde{\mathbf{C}}_n^T]^T$. Finding Vandermonde vectors $\mathbf{y}$ in the null space when $m > 1$ can be done using the eigendecomposition approach in the Sylvester method (see section 2.2). Note that in the noisefree case with $m = 1$ the starting value for $\mathbf{y}$ is of Vandermonde form $\mathrm{Vdm}(r^{-1}, d)$ and immediately reveals the common root r. The starting value for $\mathbf{z}$ in (2.11) can be taken equal to $\mathbf{M}(x)\,\mathbf{y}$, with x as in (2.8).

The initialization procedure above requires $d \leq n$. By the construction of $\mathbf{Q}$ the matrices $\widetilde{\mathbf{C}}_i$ have only their first rows nonzero for $i = 2, \ldots, n$. Hence, the nonzero rows of the matrix $[\widetilde{\mathbf{C}}_2^T \;\ldots\; \widetilde{\mathbf{C}}_n^T]^T$ form an $(n-1) \times d$ matrix. For $d > n$ the null space has dimension $d - n + 1 \geq 2$. Hence, for $m = 1$ and $d > n$ the starting value for $\mathbf{y}$ is not guaranteed to be of Vandermonde form. In that case, we adapt the procedure as follows. Let $\mathbf{N}$ be a basis of the null space of $[\widetilde{\mathbf{C}}_2^T \;\ldots\; \widetilde{\mathbf{C}}_n^T]^T$. Using the eigendecomposition approach in the Sylvester method (section 2.2), we obtain nonsingular $\mathbf{T}$ such that $\mathbf{N}\,\mathbf{T}$ contains Vandermonde vectors if they exist in the column space of $\mathbf{N}$. Next, we identify the columns of $\mathbf{N}\,\mathbf{T}$ closest to Vandermonde form, which can be used as starting values for the common eigenvector $\mathbf{y}$. For this we use the shift property of Vandermonde vectors, i.e., $\mathbf{v}(2\!:\!d) = u\,\mathbf{v}(1\!:\!d-1)$ for $\mathbf{v} = \mathrm{Vdm}(u, d)$. For a general vector $\mathbf{v}$ an estimate of generator u is obtained as $\hat{u} = \mathbf{a}^H\mathbf{b}/(\mathbf{a}^H\mathbf{a})$, with $\mathbf{a} = \mathbf{v}(1\!:\!d-1)$ and $\mathbf{b} = \mathbf{v}(2\!:\!d)$. Closeness of $\mathbf{v}$ to Vandermonde form is then assessed via $\|\mathbf{b} - \hat{u}\,\mathbf{a}\|^2$. Note that least squares estimation of a

Vandermonde generator is known to yield inaccurate results due to the fact that terms with large powers dominate for $|u| > 1$ and terms with small powers dominate for $|u| < 1$.

In the above we assumed that all polynomials $p_i(x)$ have the same degree d. For a polynomial $p_i(x)$ with degree $d_i < d$, the algorithm can be adapted as follows. Instead of $\mathbf{C}_i - x\,\mathbf{I}_{d_i}$ $(d_i \times d_i)$ in (2.7), we work with

$$\begin{bmatrix} \mathbf{C}_i & \mathbf{0} & \dots & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_i & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_i & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{C}_i \end{bmatrix} - x \begin{bmatrix} \mathbf{I}_{d_i} & \mathbf{0} & \dots & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{d_i} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{d_i} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{I}_{d_i} \end{bmatrix}, \tag{2.14}$$

which has size $(d - d_i + 1)\,d_i \times d$. We write (2.14) as $\mathbf{C}_i^{\text{alt}} - x\,\mathbf{I}_i^{\text{alt}}$, with $\mathbf{C}_i^{\text{alt}} = \mathbf{C}_i$ and $\mathbf{I}_i^{\text{alt}} = \mathbf{I}_{d_i}$ when $d_i = d$. Note that $\mathbf{C}_i^{\text{alt}}\mathbf{y} = r\,\mathbf{I}_i^{\text{alt}}\mathbf{y}$ for $\mathbf{y} = \text{Vdm}(r^{-1}, d)$, with r a common root. Let $\mathbf{C}_{\text{mat}}^{\text{alt}} = [(\mathbf{C}_1^{\text{alt}})^T \ \dots \ (\mathbf{C}_n^{\text{alt}})^T]^T$ and $\mathbf{I}_{\text{mat}}^{\text{alt}} = [(\mathbf{I}_1^{\text{alt}})^T \ \dots \ (\mathbf{I}_n^{\text{alt}})^T]^T$. Instead of $\mathbf{M}(x)$ in (2.7), we work with $\mathbf{M}^{\text{alt}}(x) = \mathbf{C}_{\text{mat}}^{\text{alt}} - x\,\mathbf{I}_{\text{mat}}^{\text{alt}}$. The least squares update (2.8) of x is replaced by

$$x = (\mathbf{y}^H(\mathbf{I}_{\text{mat}}^{\text{alt}})^H\mathbf{I}_{\text{mat}}^{\text{alt}}\mathbf{y})^{-1}\mathbf{y}^H(\mathbf{I}_{\text{mat}}^{\text{alt}})^H\mathbf{C}_{\text{mat}}^{\text{alt}}\mathbf{y}\,. \tag{2.15}$$

Since (2.15) is not of Rayleigh quotient form, we do not use method RQI-M. For methods RQ-LS-I and RQ-LS-A the matrix $\mathbf{M}(x)$ is replaced by $\mathbf{M}^{\text{alt}}(x)$ in the updates of $\mathbf{y}$ in (2.10) and (2.11), respectively. Since the companion matrices $\mathbf{C}_i$ are not all of the same size, we use our tensor-based procedure for computing initial values for a subset of polynomials that have the same degree.

3 Local minima in the Rayleigh quotient problem

We use the minimization of $\|\mathbf{M}(x)\,\mathbf{y}\|^2$ to analyze properties of local and global minima, with the aim of distinguishing between them. Let $f(x, \mathbf{y}) = \|\mathbf{M}(x)\,\mathbf{y}\|^2$. The goal is to minimize $f(x, \mathbf{y})$ subject to $\mathbf{y}^H\mathbf{y} = 1$. Let $\mathbf{C}_{\text{mat}} = [\mathbf{C}_1^T \ \dots \ \mathbf{C}_n^T]^T$ and $\mathbf{C}_{\text{sum}} = \mathbf{C}_1 + \cdots + \mathbf{C}_n$. We formulate the Lagrangian of the problem as $L(\lambda, x, \mathbf{y}) = f(x, \mathbf{y}) - \lambda\,(\mathbf{y}^H\mathbf{y} - 1)$, with Lagrange multiplier λ. We have $f(x, \mathbf{y}) = \mathbf{y}^H\mathbf{C}_{\text{mat}}^H\mathbf{C}_{\text{mat}}\mathbf{y} - 2\,x\,\mathbf{y}^H\mathbf{C}_{\text{sum}}\,\mathbf{y} + x^2\,n\,\mathbf{y}^H\mathbf{y}$. The gradient ∇L is given by

$$\nabla L = \begin{pmatrix} \frac{\delta L}{\delta \lambda} \\ \frac{\delta L}{\delta x} \\ \frac{\delta L}{\delta \mathbf{y}} \end{pmatrix} = \begin{pmatrix} 1 - \mathbf{y}^H\mathbf{y} \\ -2\,\mathbf{y}^H\mathbf{C}_{\text{sum}}\,\mathbf{y} + 2\,x\,n\,\mathbf{y}^H\mathbf{y} \\ 2\,\mathbf{C}_{\text{mat}}^H\mathbf{C}_{\text{mat}}\mathbf{y} - 2\,x\,(\mathbf{C}_{\text{sum}} + \mathbf{C}_{\text{sum}}^H)\,\mathbf{y} + 2\,x^2\,n\,\mathbf{y} - 2\,\lambda\,\mathbf{y} \end{pmatrix}. \tag{3.1}$$

In a critical point the value of λ can be computed via $\lambda = 0.5\,\mathbf{y}^H\,\frac{\delta f}{\delta \mathbf{y}}$, which implies $\frac{\delta L}{\delta \mathbf{y}} = \mathbf{0}$. Note that this also implies $\lambda = f(x, \mathbf{y})$ in the critical point (see the expression of $f(x, \mathbf{y})$ above). The

bordered Hessian $\mathbf{H}$ of L is given by

$$\begin{bmatrix} 0 & 0 & -2\,\mathbf{y}^H \\ 0 & 2\,n\,\mathbf{y}^H\mathbf{y} & -2\,\mathbf{y}^H(\mathbf{C}_{\text{sum}} + \mathbf{C}_{\text{sum}}^H) + 4\,x\,n\,\mathbf{y}^H \\ -2\,\mathbf{y} & -2\,(\mathbf{C}_{\text{sum}} + \mathbf{C}_{\text{sum}}^H)\,\mathbf{y} + 4\,x\,n\,\mathbf{y} & 2\,\mathbf{C}_{\text{mat}}^H\mathbf{C}_{\text{mat}} - 2\,x\,(\mathbf{C}_{\text{sum}} + \mathbf{C}_{\text{sum}}^H) + 2\,x^2\,n\,\mathbf{I}_d \end{bmatrix}. \quad (3.2)$$

Any local minimum is characterized by $\nabla L = \mathbf{0}$ and bordered Hessian $\mathbf{H}$ with negative principal minors of size $3, \ldots, d+2$. A global minimum is a local minimum with $f(u, \mathbf{y}) = \lambda = 0$.

Consider the following example with $n = 2$ polynomials of degree $d = 3$:

$$p_1(x) = x^3 - x\,, \qquad p_2(x) = x^3 - 1.5\,x^2 - 1.5\,x + 1\,. \quad (3.3)$$

The polynomials have one common root $r = -1$ with multiplicity one. We run the Rayleigh quotient algorithms RQI-M, RQ-LS-I, and RQ-LS-A with initial $\mathbf{y}$ as specified in section 2.4, and we stop them when the absolute change in consecutive updates of x drops below 0.0001. As expected, the algorithms need only 2 iterations to accurately obtain the common root up to machine precision. The gradient (3.1) entries are smaller than 10^{-14} in magnitude, the principal minors of size 3,4,5 of the bordered Hessian (3.2) are equal to -5.3, -291.6, and -3376, and λ is smaller than $1.3 \cdot 10^{-16}$. Hence, a clear indication that the global minimum has been found.

For random starting values $\mathbf{y}$ and $\mathbf{z}$ the RQ-LS-A algorithm terminates at the false solution $x \approx 0.5525$ in 720 out of 1000 runs. Here, we have $\nabla L \approx \mathbf{0}$, principal minors 3,4,5 of $\mathbf{H}$ approximately equal to -1.4, -33.5, and -517.0, and $\lambda \approx 0.08$. The final common eigenvector estimate is not Vandermonde. Hence, this is a local minimum. The same local minimum is found in 708 out of 1000 runs by the RQ-LS-I algorithm with random starting value $\mathbf{y}$.

For random starting value $\mathbf{y}$ the RQI-M algorithm terminates at false solutions $x \approx 1.3904$ and $x \approx 0.3596$ in 258 and 469 out of 1000 runs, respectively. For both false solutions the derivative $\partial L/\partial \mathbf{y}$ is not close to zero, which is not surprising since the RQI update of $\mathbf{y}$ does not minimize $\|\mathbf{M}(x)\,\mathbf{y}\|^2$.

For comparison, the Newton-Raphson method with random starting point and identical stopping criterion as above terminates at false solution $x \approx 0.4768$ in 680 out of 1000 runs.

4 Simulation study

We assess the performance of the numerical rootfinding methods in an extensive simulation study. For various values of degrees d_i, number n of polynomials, number m of common roots, numbers of real and complex roots, and noise level on the coefficients, we proceed as follows. We generate each noisefree polynomial $p_i(x)$ by sampling $d_i - m$ random roots (real or complex or a combination of

those). Together with the m common roots and after specifying the leading coefficient (more on this later), this defines each $p_i(x)$. Next, we generate 100 noisy polynomials systems by adding iid $N(0, \sigma^2)$ noise to all coefficients of each $p_i(x)$, for some σ. The rootfinding methods are applied to all 100 noisy systems and the mean absolute deviation (MAD) and the standard deviation of the absolute deviation (SAD) are computed for each method. For common roots $r_1, \ldots, r_m$ and estimates $\hat{r}_{ih}$, $i = 1, \ldots, 100$, $h = 1, \ldots, m$, the MAD and SAD are defined as

$$\text{MAD} = \frac{1}{100\,m} \sum_{h=1}^{m} \sum_{i=1}^{100} |\hat{r}_{ih} - r_h|\,, \qquad \text{SAD} = \sqrt{\frac{1}{100\,m} \sum_{h=1}^{m} \sum_{i=1}^{100} (|\hat{r}_{ih} - r_h| - \text{MAD})^2}\,. \quad (4.1)$$

Hence, the MAD is an average measure of estimation accuracy for all noisy systems, while the SAD represents the variation of estimation accuracy over the noisy systems. For each fixed set of d_i, n, m, σ, and combination of real and complex roots, we generate 100 noisefree systems as explained above. Hence, for each noisefree system we obtain 100 MAD and SAD values for each rootfinding method. The results are very similar when more than 100 noisefree systems are generated. Below we report the results for various cases.

Details on the use of the rootfinding methods are as follows. The stopping criteria for Newton-Raphson and the Rayleigh quotient methods RQI-M, RQ-LS-I, and RQ-LS-A are when the absolute change in the root estimate drops below 0.0001 or when the number of iterations exceeds 1000. We use the method of computing all roots of all polynomials to obtain initial values for Newton-Raphson. Concerning the Sylvester method: for $n > 2$ we use the Sylvester matrix of smallest degree d_s such that (2.5) holds for the noisy system. For RQ-LS-A we compute the QR-decomposition of $\mathbf{M}(x)$ in each iteration. More efficient implementations can be achieved by using the same $\mathbf{Q}$ for several iterations and making use of the sparsity of $\mathbf{M}(x)$. However, since RQ-LS-A is outperformed by RQ-LS-I in terms of accuracy in all simulations below, we have not studied this further.

4.1 Results for $m = 1$ common root and equal degrees $d_i = d$

As common root we set $r = -1$. The real roots of noisefree $p_i(x)$ are sampled uniformly from the interval $(-1 - b, -1 + b)$. The complex roots of noisefree $p_i(x)$ are sampled uniformly from the circle in $\mathbb{C}^2$ with center -1 and radius b. The number of real roots is taken equal for all $p_i(x)$. We set the leading coefficient of $p_i(x)$ equal to one. In Table 1 the mean and standard deviation of the MAD and SAD values are given for various values of degree d, number of polynomials n, and radius b. We set noise level $\sigma = 0.3$ and sample only real roots for noisefree $p_i(x)$, Also given are the number of noisefree systems for which each method has the smallest MAD value.

	$(d, n, \sigma, \text{real}, \text{radius})$	mean MAD	std MAD	mean SAD	std SAD	best MAD
N-R	(6, 10, 0.3, 6, 3)	0.0396	0.0660	0.0438	0.1281	92
Sylvester	(6, 10, 0.3, 6, 3)	0.3440	0.1651	0.4292	0.2418	0
RQI-M	(6, 10, 0.3, 6, 3)	0.1167	0.0457	0.1562	0.0815	0
RQ-LS-I	(6, 10, 0.3, 6, 3)	0.0870	0.0351	0.1384	0.0968	8
RQ-LS-A	(6, 10, 0.3, 6, 3)	0.1606	0.0414	0.1457	0.0570	0
N-R	(6, 10, 0.3, 6, 2)	0.2815	0.1947	0.1101	0.1200	28
Sylvester	(6, 10, 0.3, 6, 2)	0.3206	0.0750	0.3381	0.1219	2
RQI-M	(6, 10, 0.3, 6, 2)	0.2422	0.0673	0.1948	0.0519	7
RQ-LS-I	(6, 10, 0.3, 6, 2)	0.1847	0.0531	0.1539	0.0517	52
RQ-LS-A	(6, 10, 0.3, 6, 2)	0.2202	0.0489	0.1621	0.0398	11
N-R	(6, 10, 0.3, 6, 1)	0.5927	0.1051	0.0970	0.0233	1
Sylvester	(6, 10, 0.3, 6, 1)	0.4520	0.0492	0.2936	0.0423	16
RQI-M	(6, 10, 0.3, 6, 1)	0.4800	0.0700	0.2583	0.0420	6
RQ-LS-I	(6, 10, 0.3, 6, 1)	0.4072	0.0591	0.2691	0.0427	72
RQ-LS-A	(6, 10, 0.3, 6, 1)	0.4423	0.0495	0.2508	0.0268	5
N-R	(6, 40, 0.3, 6, 1)	0.5160	0.0608	0.0373	0.0060	0
Sylvester	(6, 40, 0.3, 6, 1)	0.3549	0.0341	0.2475	0.0365	34
RQI-M	(6, 40, 0.3, 6, 1)	0.4049	0.0431	0.2251	0.0288	3
RQ-LS-I	(6, 40, 0.3, 6, 1)	0.3420	0.0371	0.2096	0.0256	60
RQ-LS-A	(6, 40, 0.3, 6, 1)	0.3698	0.0353	0.1965	0.0205	3
N-R	(15, 20, 0.3, 15, 2)	0.5274	0.4052	0.2227	0.1983	10
Sylvester	(15, 20, 0.3, 15, 2)	0.3135	0.0525	0.2456	0.0460	1
RQI-M	(15, 20, 0.3, 15, 2)	0.2027	0.0715	0.1553	0.0541	22
RQ-LS-I	(15, 20, 0.3, 15, 2)	0.1769	0.0590	0.1383	0.0569	67
RQ-LS-A	(15, 20, 0.3, 15, 2)	0.2904	0.0532	0.1968	0.0343	0

Table 1: Results for polynomial systems with $m = 1$ common root and equal degrees $d_i = d$, for 100 generated noisefree polynomial systems and 100 noisy replications each. MAD and SAD values are computed per noisefree system.

The results in Table 1 show that smaller radius yields larger MAD values, which is as expected. For relatively large radius b the Newton-Raphson method is most accurate, while for smaller radius the RQ-LS-I is most accurate followed by the RQ-LS-A and Sylvester methods. The difference in accuracy for small radius is larger than for large radius. Increasing the number n of polynomials does not change the performance differences when the radius is small. For $d = 6$ the Newton-Raphson method has smaller mean SAD values than both Sylvester and the RQ methods, indicating that estimation accuracy does not vary as much over the noisy replications of each noisefree system. Of the two least RQ methods RQ-LS-I has smaller mean MAD over all systems in all cases, but for

some systems RQ-LS-A has a smaller mean MAD. In general the performance of RQ-LS-I is better.

The results in Table 1 only include polynomials with real roots. However, the results are very similar when also complex roots are considered. The same is true for varying the degree d, number of polynomials n, and the noise level σ. Hence, the main difference in performance between the methods in the current simulations is caused by varying the radius of the circle or line from which the random roots are sampled.

We expect the estimation of the common roots to be more difficult when there exists a set of other roots, one for each noisefree polynomial, that are close together. To check whether this is true we need to compute the smallest standard deviation of such a set of roots for the noisefree system. Since there are $\prod_i (d_i - m)$ such subsets to check, we use the heuristic method explained in section 2.3. The resulting standard deviation is denoted as σ_{roots} and corresponds to the noisefree system.

The influence of σ_{roots} on the performance of the methods can be seen in Figure 1, of which the right plot corresponds to simulations done in section 4.3. As can be seen, in the left plot only the Newton-Raphson method has larger MAD values when σ_{roots} is smaller, while in the right plot this also holds for the RQ methods. The accuracy of the Sylvester method is less affected by small σ_{roots}.

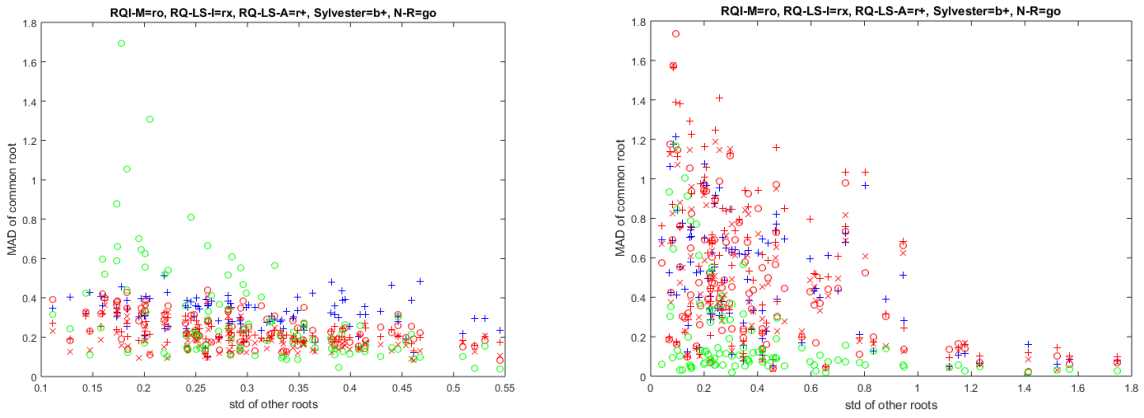


Figure 1: Mean MAD values versus σ_{roots} for $d_i = 6$, $n = 10$, $\sigma = 0.3$, only real roots, and radius $b = 2$ (left), and for $d_i = 4$, $n = 3$, $\sigma = 0.2$, only real roots, and radius $b = 3$ (right), for 100 generated noisefree polynomial systems and 100 noisy replications each. Methods are Newton-Raphson (green circle), Sylvester (blue plus), RQI-M (red circle), RQ-LS-I (red cross), and RQ-LS-A (red plus).

4.2 Varying the derivative at the common root

Here we run simulations with the same setup as in section 4.1, except we choose the leading coefficient of each $p_i(x)$ in a different way. We expect that finding a noisy common root will be more difficult when the derivative of each noisefree $p_i(x)$ is small in the neighborhood of the common root. Small perturbations of $p_i(x)$ may then result in relatively large changes of the common root. To test this hypothesis and compare the performance of the rootfinding methods, we generate the noisefree polynomials $p_i(x)$ as follows. We write $p_i(x) = (x - r)\, k_i(x)$, with r denoting the common root. Then $\partial p_i/\partial x$ evaluated at $x = r$ equals $k_i(r)$. First, polynomials $k_i(x)$ are generated by randomly sampling $d - 1$ roots as in section 4.1, with leading coefficient set to one. Next, the coefficients of $k_i(x)$ are multiplied by $w/k_i(r)$ such that $k_i(r)$ has a desired value w. This completely specifies $p_i(x)$.

As above, we set $r = -1$ as common root. We focus on the effect of varying the derivatives at the common root while keeping other parameters constant, with $d = 6$, $n = 10$, $\sigma = 0.1$, only real roots, and radius $b = 3$. For each noisefree $p_i(x)$ we sample the size of the derivative at the common root uniformly from the interval $(-\alpha, +\alpha)$. For various values of α the results are summarized in Table 2. As can be seen, when α decreases the mean MAD values increase for all methods. Hence, small derivatives at the common root indeed make it more difficult to obtain accurate estimates. The comparison of the methods is as follows. The RQI-M and especially RQ-LS-I methods are best for small values of α, while Newton-Raphson is preferred for large α. As in Table 1, Newton-Raphson has smaller mean SAD values than both the Sylvester and RQ methods. Also, RQ-LS-I again outperforms RQ-LS-A in terms of mean MAD values.

4.3 The case $d > n$

As explained at the end of section 2.4, the initialization procedure for the RQ methods requires $d \leq n$ to obtain a Vandermonde vector as starting value for the common eigenvector $\mathbf{y}$ in the noisefree case. However, we proposed an adaptation to the case $d > n$ by using the eigendecomposition approach in the Sylvester method (section 2.2). Here, we evaluate this adapted initialization procedure. For some d and n with $d > n$ we generate 100 noisefree systems as in section 4.1, with only real roots and radius $b = 3$. In Table 3 the mean MAD is given for each rootfinding method, where the mean is now taken over the 100 root estimates (one for each generated system). We use the RQ methods with both the original and the adapted initialization procedure. The Sylvester method is applied with degree d_s of the generalized Sylvester matrix (2.4) such that it is a tall or square matrix (i.e., (2.5) holds). Note that taking $d_s = d$ in (2.5) implies $n - 1 \geq d$. Hence, when $d > n$ the degree d_s of the generalized Sylvester matrix should be taken larger than d.

	α	mean MAD	std MAD	mean SAD	std SAD	best MAD
N-R	5	0.0289	0.0143	0.0295	0.0515	96
Sylvester	5	0.1809	0.1534	0.2122	0.2195	4
RQI-M	5	0.1993	0.0885	0.2716	0.1522	0
RQ-LS-I	5	0.1787	0.0962	0.2398	0.1581	0
RQ-LS-A	5	0.2272	0.1042	0.2395	0.1322	0
N-R	0.5	0.1911	0.1113	0.0807	0.1321	80
Sylvester	0.5	0.3230	0.0846	0.3149	0.1261	7
RQI-M	0.5	0.3129	0.0552	0.2649	0.0745	1
RQ-LS-I	0.5	0.2588	0.0432	0.2055	0.0631	12
RQ-LS-A	0.5	0.3097	0.0615	0.2385	0.0609	0
N-R	0.05	0.4757	0.1975	0.1450	0.1125	42
Sylvester	0.05	0.5241	0.1257	0.3533	0.0872	7
RQI-M	0.05	0.4925	0.1106	0.3723	0.1014	9
RQ-LS-I	0.05	0.4527	0.1136	0.3147	0.0885	42
RQ-LS-A	0.05	0.5346	0.1141	0.3076	0.0573	0
N-R	0.01	0.6721	0.1868	0.1910	0.1227	28
Sylvester	0.01	0.6868	0.1238	0.3683	0.0436	7
RQI-M	0.01	0.6324	0.1396	0.4573	0.1090	29
RQ-LS-I	0.01	0.6183	0.1378	0.3990	0.0758	36
RQ-LS-A	0.01	0.6957	0.1223	0.3400	0.0342	0

Table 2: Results for polynomial systems with $m = 1$ common root, $d_i = 6$, $n = 10$, $\sigma = 0.1$, only real roots, and radius $b = 3$, for 100 generated noisefree polynomial systems and 100 noisy replications each. The size of the derivative at the common root is sampled from $(-\alpha, +\alpha)$. MAD and SAD values are computed per noisefree system.

	(d, n)	mean MAD	(d, n)	mean MAD	(d, n)	mean MAD
N-R	(4,3)	$1.4 \cdot 10^{-16}$	(6,5)	$3.8 \cdot 10^{-16}$	(20,10)	$6.3 \cdot 10^{-12}$
Sylvester	(4,3)	$2.7 \cdot 10^{-14}$	(6,5)	$5.4 \cdot 10^{-14}$	(20,10)	$1.0 \cdot 10^{-4}$
RQI-M (original)	(4,3)	0.1090	(6,5)	1.0381	(20,10)	1.9797
RQ-LS-I (original)	(4,3)	0.0654	(6,5)	1.0885	(20,10)	5.7899
RQ-LS-A (original)	(4,3)	0.4067	(6,5)	1.2289	(20,10)	7.1350
RQI-M (adapted)	(4,3)	$1.0 \cdot 10^{-15}$	(6,5)	$2.4 \cdot 10^{-14}$	(20,10)	$2.5 \cdot 10^{-11}$
RQ-LS-I (adapted)	(4,3)	$4.8 \cdot 10^{-15}$	(6,5)	$2.6 \cdot 10^{-13}$	(20,10)	0.0173
RQ-LS-A (adapted)	(4,3)	$3.6 \cdot 10^{-15}$	(6,5)	$1.7 \cdot 10^{-13}$	(20,10)	$1.3 \cdot 10^{-9}$

Table 3: Results for noisefree polynomial systems with $m = 1$ common root, equal degrees $d_i = d$, only real roots, and radius $b = 3$. Mean MAD values are computed for 100 generated systems.

As can be seen in Table 3, all RQ methods fail to find the common root when the original initialization is used, but the adapted procedure yields accurate results. When the difference between d and n is relatively large, however, the results are somewhat less accurate for RQ-LS-I.

In Table 4 we report the results for noisy systems with the same values of d and n as in Table 3. For radius $b = 2$ the Newton-Raphson method suffers from large variation in the MAD values, while the RQI-M, RQ-LS-I, and Sylvester methods are quite stable. As in section 4.1, the Newton-Raphson methods perform better on average than the RQ methods when the radius b is increased, and RQ-LS-I performs better than RQ-LS-A. Interestingly, for $d = 20$, $n = 10$, and radius $b = 2$ the Sylvester method performs best by far. Apparently, increasing the degree d_s of the generalized Sylvester matrix yields a very robust method when $d >> n$ and the radius b is relatively small.

4.4 Results for $m = 2$ common roots and equal degrees $d_i = d$

Here we generate polynomials with $m = 2$ common roots: $r_1 = -1$ and $r_2 = +1$. The noisefree polynomials $p_i(x)$ are generated analogous to section 4.1, with the real roots sampled uniformly from $(-b, +b)$, and the complex roots sampled uniformly from the circle in $\mathbb{C}^2$ with center 0 and radius b. For all methods we assume the number of $m = 2$ common roots to be known. For Newton-Raphson we compute two initial values, one for each common root, and run the algorithm for each initial value separately. Initial values obtained by computing all roots of all $p_i(x)$ (section 2.3) correspond to the set of roots (one for each $p_i(x)$) with the smallest standard deviation, and the set of roots with the smallest standard deviations when the roots of the first set have been removed. For the RQ methods $m = 2$ initial values are computed as described at the end of section 2.4, and we run each RQ algorithm for each initial value separately.

In Table 5 the results are reported for some of the same values of d, n, and radius b as in Table 1. As can be seen, the mean MAD values are smaller than in Table 1 except for the Sylvester and RQ methods when $d = 15$. Again, smaller radius b is in favor of the RQ and Sylvester methods as for $m = 1$. As in the simulation results above, RQ-LS-I outperforms RQ-LS-A.

	$(d, n, \sigma, \text{real}, \text{radius})$	mean MAD	std MAD	mean SAD	std SAD	best MAD
N-R	(4, 3, 0.2, 4, 2)	0.3840	0.2787	0.2678	0.1813	56
Sylvester	(4, 3, 0.2, 4, 2)	0.4065	0.1292	0.3979	0.1709	22
RQI-M	(4, 3, 0.2, 4, 2)	0.4486	0.1271	0.4576	0.1675	1
RQ-LS-I	(4, 3, 0.2, 4, 2)	0.4158	0.1377	0.4434	0.1943	17
RQ-LS-A	(4, 3, 0.2, 4, 2)	0.5541	0.2278	0.5197	0.2827	4
N-R	(4, 3, 0.2, 4, 3)	0.2027	0.2424	0.2806	0.2784	78
Sylvester	(4, 3, 0.2, 4, 3)	0.4963	0.2829	0.5287	0.3200	4
RQI-M	(4, 3, 0.2, 4, 3)	0.4922	0.3351	0.6205	0.3763	4
RQ-LS-I	(4, 3, 0.2, 4, 3)	0.4755	0.3343	0.5832	0.3853	10
RQ-LS-A	(4, 3, 0.2, 4, 3)	0.5951	0.3803	0.6400	0.4326	4
N-R	(6, 5, 0.2, 6, 2)	0.4765	0.4103	0.2374	0.1997	46
Sylvester	(6, 5, 0.2, 6, 2)	0.4064	0.1180	0.4445	0.1745	25
RQI-M	(6, 5, 0.2, 6, 2)	0.4063	0.0943	0.4606	0.1585	5
RQ-LS-I	(6, 5, 0.2, 6, 2)	0.3608	0.0808	0.3994	0.1428	24
RQ-LS-A	(6, 5, 0.2, 6, 2)	0.5299	0.1425	0.5036	0.2126	0
N-R	(6, 5, 0.2, 6, 3)	0.1287	0.2292	0.1840	0.2991	85
Sylvester	(6, 5, 0.2, 6, 3)	0.5546	0.2775	0.6183	0.3202	6
RQI-M	(6, 5, 0.2, 6, 3)	0.5114	0.2805	0.7286	0.3447	3
RQ-LS-I	(6, 5, 0.2, 6, 3)	0.4723	0.2845	0.6326	0.3314	6
RQ-LS-A	(6, 5, 0.2, 6, 3)	0.6589	0.3053	0.7344	0.3877	0
N-R	(20, 10, 0.2, 20, 2)	0.9377	0.4614	0.3219	0.1924	3
Sylvester	(20, 10, 0.2, 20, 2)	0.2994	0.0499	0.3296	0.0928	96
RQI-M	(20, 10, 0.2, 20, 2)	0.6401	0.1663	0.7455	0.2418	1
RQ-LS-I	(20, 10, 0.2, 20, 2)	0.7227	0.1858	0.8066	0.3825	0
RQ-LS-A	(20, 10, 0.2, 20, 2)	0.8522	0.2179	1.1167	1.0193	0
N-R	(20, 10, 0.2, 20, 3)	0.1747	0.3061	0.2416	0.2988	91
Sylvester	(20, 10, 0.2, 20, 3)	0.9477	0.2989	1.1361	0.3307	3
RQI-M	(20, 10, 0.2, 20, 3)	0.8447	0.3197	0.7685	0.3289	5
RQ-LS-I	(20, 10, 0.2, 20, 3)	0.9999	0.4244	0.9439	0.4535	0
RQ-LS-A	(20, 10, 0.2, 20, 3)	1.0073	0.4254	0.9755	0.5253	1

Table 4: Results for polynomial systems with $m = 1$ common root and equal degrees $d_i = d > n$, for 100 generated noisefree polynomial systems and 100 noisy replications each. MAD and SAD values are computed per noisefree system.

	$(d, n, \sigma, \text{real}, \text{radius})$	mean MAD	std MAD	mean SAD	std SAD	best MAD
N-R	(6, 10, 0.3, 6, 3)	0.0168	0.0263	0.0296	0.0604	97
Sylvester	(6, 10, 0.3, 6, 3)	0.0882	0.0752	0.0961	0.1067	0
RQI-M	(6, 10, 0.3, 6, 3)	0.0646	0.0352	0.0917	0.0662	0
RQ-LS-I	(6, 10, 0.3, 6, 3)	0.0440	0.0185	0.0642	0.0764	3
RQ-LS-A	(6, 10, 0.3, 6, 3)	0.0789	0.0480	0.0808	0.0697	0
N-R	(6, 10, 0.3, 6, 2)	0.1530	0.1079	0.2541	0.1314	27
Sylvester	(6, 10, 0.3, 6, 2)	0.1390	0.1020	0.1514	0.1249	14
RQI-M	(6, 10, 0.3, 6, 2)	0.1648	0.0929	0.2003	0.1212	0
RQ-LS-I	(6, 10, 0.3, 6, 2)	0.1118	0.1014	0.1914	0.1615	59
RQ-LS-A	(6, 10, 0.3, 6, 2)	0.1533	0.1078	0.1683	0.1142	0
N-R	(15, 20, 0.3, 15, 2)	0.2437	0.1299	0.3910	0.1345	82
Sylvester	(15, 20, 0.3, 15, 2)	0.4572	0.0859	0.4049	0.0574	0
RQI-M	(15, 20, 0.3, 15, 2)	0.3907	0.1021	0.4944	0.0750	6
RQ-LS-I	(15, 20, 0.3, 15, 2)	0.3972	0.1247	0.5607	0.0901	12
RQ-LS-A	(15, 20, 0.3, 15, 2)	0.4704	0.0913	0.3931	0.0509	0
N-R	(15, 20, 0.3, 15, 1)	0.9995	0.0098	0.2399	0.0218	0
Sylvester	(15, 20, 0.3, 15, 1)	0.7636	0.0221	0.2983	0.0142	2
RQI-M	(15, 20, 0.3, 15, 1)	0.6643	0.0472	0.5835	0.0288	98
RQ-LS-I	(15, 20, 0.3, 15, 1)	0.7743	0.0348	0.6178	0.0191	0
RQ-LS-A	(15, 20, 0.3, 15, 1)	0.7749	0.0227	0.2965	0.0135	0

Table 5: Results for polynomial systems with $m = 2$ common roots and equal degrees $d_i = d$, for 100 generated noisefree polynomial systems and 100 noisy replications each. MAD and SAD values are computed per noisefree system and for the estimates of both common roots combined.

4.5 Unequal degrees d_i

Here we evaluate the performance of the rootfinding methods when the n polynomials $p_i(x)$ do not have equal degree. As illustrative example we take $n = 10$, $d_i = 6, 6, 6, 5, 5, 4, 4, 3, 3, 3$, $\sigma = 0.3$, radiuses $b = 2$ and $b = 1$, $m = 2$ common roots $r_1 = -1$ and $r_2 = +1$, and the number of real roots equal to $4, 4, 4, 4, 4, 3, 3, 2, 2, 2$. Hence, each polynomial also has some complex roots. Newton-Raphson can be applied as outlined in section 2.1. For the Sylvester method, the degree d_s should satisfy (2.5). We take $d_s = 4$, which is the smallest d_s satisfying (2.5). As stated at the end of section 2.4, to be able to apply the RQ methods we replace matrix $\mathbf{M}(x)$ in (2.7) by $\mathbf{M}^{\text{alt}}(x)$ and use only methods RQ-LS-I and RQ-LS-A. Initial values are computed based on the last three polynomials of degree 3. The results are reported in Table 6 below.

	$(n, \sigma, \text{radius})$	mean MAD	std MAD	mean SAD	std SAD	best MAD
N-R	(10, 0.3, 2)	0.0656	0.0301	0.0676	0.0496	78
Sylvester	(10, 0.3, 2)	0.0995	0.0211	0.0713	0.0202	6
RQ-LS-I	(10, 0.3, 2)	0.0878	0.0219	0.1489	0.0766	16
RQ-LS-A	(10, 0.3, 2)	0.2895	0.0970	0.3921	0.6780	0
N-R	(10, 0.3, 1)	0.4964	0.1245	0.4230	0.0402	0
Sylvester	(10, 0.3, 1)	0.2225	0.0713	0.1876	0.0733	68
RQ-LS-I	(10, 0.3, 1)	0.2704	0.0893	0.3680	0.0806	10
RQ-LS-A	(10, 0.3, 1)	0.2704	0.1160	0.3840	0.8238	22

Table 6: Results for polynomial systems with $m = 2$ common roots and unequal degrees $d_i = 6, 6, 6, 5, 5, 4, 4, 3, 3, 3$, for 100 generated noisefree polynomial systems and 100 noisy replications each. MAD and SAD values are computed per noisefree system and for the estimates of both common roots combined.

As can be seen, the RQ methods perform better for smaller radius, yet are outperformed by the Sylvester method for small radius. In contrast, Newton-Raphson is the best method for larger radius. As in the other simulations, RQ-LS-A is outperformed by RQ-LS-I.

5 Discussion

In this paper we have introduced new Rayleigh quotient based methods for estimating common roots of a noisy system of univariate polynomials. Using a tensor perspective on the companion matrices of the polynomials, we are able to compute reasonable starting values for the RQ methods. Among the three RQ methods the performance of RQ-LS-I is best, i.e., when the common root is

estimated via the Rayleigh quotient and the common eigenvector is estimated by inverse iteration. RQ-LS-I is derived from least squares minimization of each $\widetilde{\mathbf{C}}_i - x\,\mathbf{I}_d$, with $\widetilde{\mathbf{C}}_i$ the noisy version of the ith companion matrix. Compared to the Newton-Raphson and Sylvester methods, RQ-LS-I is to be preferred when the other roots of the (noisefree) polynomials are relatively close to the common root(s), or when the derivatives of the (noisefree) polynomials at the common root(s) are very small in magnitude. In both cases it is difficult to obtain good estimates of the common root(s) when noise is added to the system. We have demonstrated that the RQ methods work well in general, also when the degree d exceeds the number n of polynomials, when there are multiple common roots to be estimated, and when the polynomials in the system have unequal degrees.

Apart from the methods considered in the simulation study, a group of numerical methods exist for computing an approximate greatest common divisor (AGCD) of a set of perturbed univariate polynomials, e.g. [11, 10] and the references therein. Estimates of the common roots can then be obtained as roots of the AGCD. However, these methods assume only very small perturbations of the polynomials and most of them only consider $n = 2$ polynomials. Therefore, we have not included them in the simulation study.

As a topic for further study the backward stability (in terms of the common roots) of the RQ methods may be analysed for the noisefree case. In [7] it is shown that for a single noisefree univariate polynomial and a backward stable method to compute the eigenvalues of the companion matrix, backward stability of the rootfinding method depends on the largest absolute value of the polynomial coefficients after dividing by the leading coefficient. It may be investigated whether this approach can be extended to finding a common root of multiple univariate polynomials via their companion matrices. As in [4] alternative companion matrix forms such as Fiedler matrices may yield better stability properties in some cases.

References

[1] S. Barnett, Polynomials and Linear Control Systems, Marcel Dekker, Inc., New York, 1983.

[2] K. Batselier, P. Dreesen, and B. De Moor, On the null spaces of the Macaulay matrix, Linear Algebra Appl., 460 (2014), pp. 259–289.

[3] D.A. Cox, J.B. Little, and D. O'Shea, Using Algebraic Geometry, 2nd edn., Springer-Verlag, New York, 2005.

[4] F. De Terán, F.M. Dopico, and J. Pérez, Backward stability of polynomial root-finding using Fiedler companion matrices, IMA J. Numer. Anal., 36 (2016), 133-173.

[5] A. Dickenstein and I.Z. Emiris, eds., Solving Polynomial Equations, vol. 14 of Algorithms and Computation in Mathematics. Springer, 2005.

[6] P. Dreesen, Back to the Roots - Polynomial System Solving Using Linear Algebra, PhD Thesis, KU Leuven, Belgium, 2013.

[7] A. Edelman and H. Murakami, Polynomial roots from companion matrix eigenvalues, Math. Comput., 64 (1995), 763-776.

[8] E. Isaacson and H.B. Keller, Analysis of Numerical Methods, John Wiley & Sons, New York, 1966

[9] C. Gotsman and S. Toledo, On the computation of null spaces of sparse rectangular matrices, SIAM J. Matrix Anal. Appl., 30 (2008), 445–463.

[10] G. Halikias, G. Galanis, N. Karcanias, and E. Milonidis, Nearest common root of polynomials, approximate greatest common divisor and the structured singular value, IMA J. Math. Contr. Inform., 30 (2013), 423–442.

[11] N. Karcanias, S. Fatouros, M. Mitrouli, and G.H. Halikias, Approximate greatest common divisor of many polynomials, generalised resultants, and strength of approximation, Comput. Math. Appl., 51 (2006), 1817–1830.

[12] D. Lazard, Résolution des systèmes d`équations algèbriques, Theor. Comput. Sci., 15 (1981), 77-110.

[13] D. Lazard, Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations, in: Computer Algebra, vol. 162 of Lecture Notes in Computer Science (J. van Hulzen, ed.), Springer Berlin, Heidelberg, 1983, pp. 146-156.

[14] F.S. Macaulay, On some formulae in elimination, Proc. London Math. Soc., 35 (1902), 3-27.

[15] F.S. Macaulay, The Algebraic Theory of Modular Systems, Cambridge University Press, 1916.

[16] V.Y. Pan and A-L. Zheng, New progress in real and complex polynomial root-finding, Compu. Math. Appl., 61 (2011), 1305–1334.

[17] H. Schwetlick and U. Schnabel, Iterative computation of the smallest singular value and the corresponding singular vectors of a matrix. Linear Algebra Appl., 371 (2003), 1–30.

[18] H.J. Stetter, Matrix eigenproblems are at the heart of polynomial system solving, ACM SIGSAM Bull., 30 (1996), 22-25.

[19] H.J. Stetter, Numerical Polynomial Algebra, SIAM, Philadelphia, 2004.

[20] J.J. Sylvester, On a theory of syzygetic relations of two rational integral functions, comprising an application to the theory of Sturms function and that of the greatest algebraical common measure. Philos. Trans. R. Soc. Lond., 143 (1853), 407–548.

[21] L.N. Trefethen and D. Bau III, Numerical Linear Algebra, Society for Industrial and Applied Mathematics, Vol. 50, 1997